Vision 3D artificielle
Session 5: Disparity maps, correlation

Pascal Monasse monasse@imagine.enpc.fr

IMAGINE, École des Ponts ParisTech

http://imagine.enpc.fr/~monasse/Stereo/

Contents

Contents

Triangulation

- Let us write again the binocular formulae:

$$\lambda x = K(RX + T) \quad \lambda' x' = K'X$$

- Write $Y^T = \begin{pmatrix} X^T & \lambda & \lambda' \end{pmatrix}$:

$$\begin{pmatrix} KR & -x & 0_3 \\ K' & 0_3 & -x' \end{pmatrix} Y = \begin{pmatrix} KT \\ 0_3 \end{pmatrix}$$

(6 equations↔5 unknowns+1 epipolar constraint)
- We can then recover X.
- Special case: $R = Id$, $T = Be_1$
- We get:

$$z(x - KK'^{-1}x') = \begin{pmatrix} Bf & 0 & 0 \end{pmatrix}^T$$

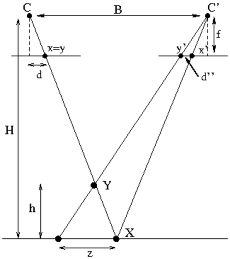
- If also $K = K'$,

$$z = fB/[(x - x') \cdot e_1] = fB/d$$

- d is the disparity

Triangulation

Fundamental principle of stereo vision



$$h \backsimeq \frac{z}{B/H}, \quad z = d'' \frac{H}{f}.$$

f focal length.
H distance optical center-ground.
B distance between optical centers (baseline).

Goal

Given two rectified images, point correspondences and computation of their apparent shift (disparity) gives information about relative depth of the scene.

Recovery of R and T

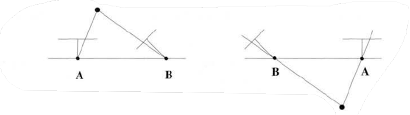
- Suppose we know K, K', and F or E. Recover R and T?
- From $E = [T]_\times R$,

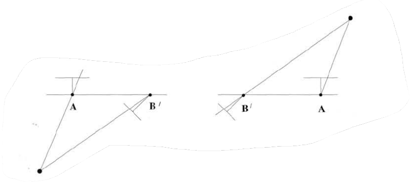
$$E^T E = R^T(TT^T - \|T\|^2 I)R = (R^T T)(R^T T)^T - \|R^T T\|^2 I$$

- If $x = R^T T$, $E^T E x = 0$ and if $y \cdot x = 0$, $E^T E y = -\|T\|^2 y$.
- Therefore $\sigma_1 = \sigma_2$ and $\sigma_3 = 0$.
- Inversely, from $E = U diag(\sigma, \sigma, 0) V^T$, we can write:

$$E = \sigma U \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} U^T U \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} V = \sigma [T]_\times R$$

- Actually, there are up to 4 solutions:





Source: Hartley-Zisserman (2003)

What is possible without calibration?

- We can recover F, but not E.
- Actually, from

  $$x = PX \quad x' = P'X$$

  we see that we have also:

  $$x = (PH^{-1})(HX) \quad x' = (P'H^{-1})(HX)$$

- Interpretation: applying a space homography and transforming the projection matrices (this changes K, K', R and T), we get exactly the same projections.
- Consequence: in the uncalibrated case, reconstruction can only be done modulo a 3D space homography.

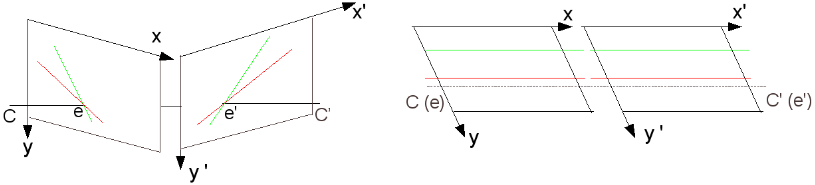
Contents

Epipolar rectification

- It is convenient to get to a situation where epipolar lines are parallel and at same ordinate in both images.
- As a consequence, epipoles are at horizontal infinity:

$$e = e' = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

- It is always possible to get to that situation by virtual rotation of cameras (application of homography)



- Image planes coincide and are parallel to baseline.

Epipolar rectification

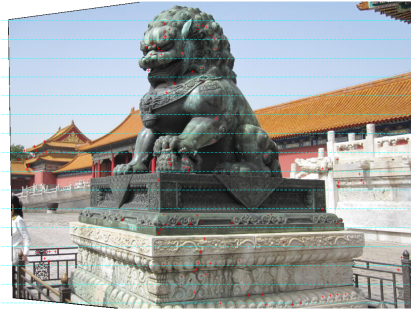
Image 1

Epipolar rectification

Image 2

Epipolar rectification

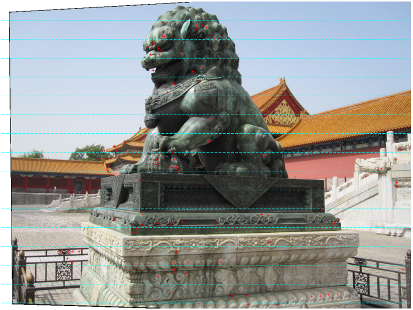
Image 1



Rectified image 1

Epipolar rectification

Image 2



Rectified image 2

Epipolar rectification

- Fundamental matrix can be written:

$$F = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}_\times = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \text{ thus } x^T F x' = 0 \Leftrightarrow y - y' = 0$$

- Writing matrices $P = K \begin{pmatrix} I & 0 \end{pmatrix}$ and $P' = K' \begin{pmatrix} I & e_1 \end{pmatrix}$:

$$K = \begin{pmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \quad K' = \begin{pmatrix} f'_x & s' & c'_x \\ 0 & f'_y & c'_y \\ 0 & 0 & 1 \end{pmatrix}$$

$$F = K^{-T} [e_1]_\times K'^{-1} = \frac{1}{f_y f_{y'}} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -f_y \\ 0 & f'_y & c'_y f_y - c_y f'_y \end{pmatrix}$$

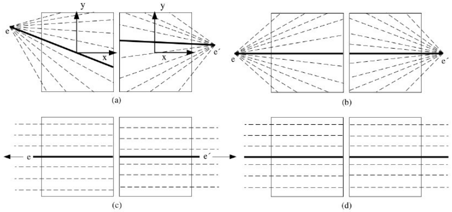
- We must have $f_y = f'_y$ and $c_y = c'_y$, that is identical second rows of K and K'

Epipolar rectification

- We are looking for homographies H and H' to apply to images such that
$$F = H^T [e_1]_\times H'$$
- That is 9 equations and 16 variables, 7 degrees of freedom remain: the first rows of K and K' and the rotation angle around baseline α
- Invariance through rotation around baseline:

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix}^T \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix} = [e_1]_\times$$

- Several methods exist, they try to distort as little as possible the image



Rectif. of Gluckman-Nayar (2001)

Epipolar rectification of Fusiello-Irsara (2008)

- We are looking for H and H' as rotations, supposing $K = K'$ known:
$$H = K_n R K^{-1} \text{ and } H' = K_n' R' K^{-1}$$
with K_n and K_n' of identical second row, R and R' rotationmatrices parameterized by Euler angles and
$$K = \begin{pmatrix} f & 0 & w/2 \\ 0 & f & h/2 \\ 0 & 0 & 1 \end{pmatrix}$$
- Writing $R = R_x(\theta_x) R_y(\theta_y) R_z(\theta_z)$ we must have:
$$F = (K_n R K^{-1})^T [e_1]_\times (K_n R' K{-}1) = K^{-T} R_z^T R_y^T [e_1]_\times R' K^{-1}$$
- We minimize the sum of squares of points to their epipolar line accoring to the 6 parametesr
$$(\theta_y, \theta_z, \theta_x', \theta_y', \theta_z', f)$$

Ruins

$\|E_0\| = 3.21$ pixels.

$\|E_6\| = 0.12$ pixels.

Ruins

$\|E_0\| = 3.21$ pixels.

$\|E_6\| = 0.12$ pixels.

Cake

$\|E_0\| = 17.9$ pixels.

$\|E_{13}\| = 0.65$ pixels.

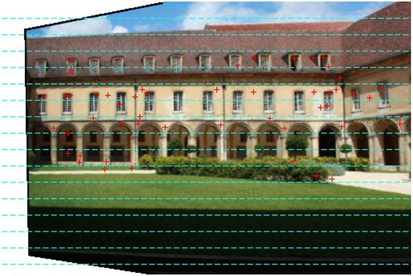
Cake

$\|E_0\| = 17.9$ pixels.

$\|E_{13}\| = 0.65$ pixels.

Cluny

$\|E_0\| = 4.87$ pixels.



$\|E_{14}\| = 0.26$ pixels.

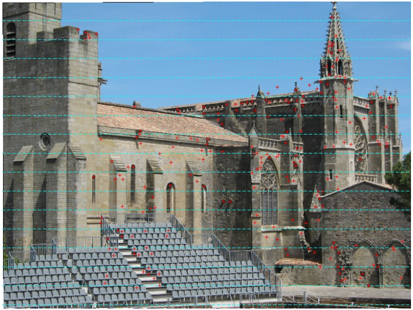
Cluny

$\|E_0\| = 4.87$ pixels.

$\|E_{14}\| = 0.26$ pixels.

Carcassonne

$\|E_0\| = 15.6$ pixels.



$\|E_4\| = 0.24$ pixels.

Carcassonne

$\|E_0\| = 15.6$ pixels.

$\|E_4\| = 0.24$ pixels.

Books

$\|E_0\| = 3.22$ pixels.

$\|E_{14}\| = 0.27$ pixels.

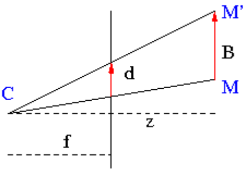
Books

$\|E_0\| = 3.22$ pixels.

$\|E_{14}\| = 0.27$ pixels.

Contents

Disparity map



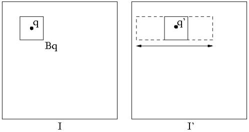
$$z = \frac{fB}{d}$$

Depth z is inversely proportional to disparity d (apparent motion, in pixels).

- Disparity map: At each pixel, its apparent motion between left and right images.
- We already know disparity at feature points, this gives an idea about min and max motion, which makes the search for matching points less ambiguous and faster.

Local search

- At each pixel, we consider a context window and we look for the motion of this window.



- Distance between windows:

$$d(q) = \arg\min_d \sum_{p \in F} (I(q+p) - I'(q + de_1 + p))^2$$

- Variants to be more robust to illumination changes:
  1. Translate intensities by the mean over the window.

$$I(q+p) \rightarrow I(q+p) - \sum_{r \in F} I(q+r)/\#F$$

  2. Normalize by mean and variance over window.

Distance between patches

Several distances or similarity measures are popular:

- SAD: Sum of Absolute Differences

$$d(q) = \arg\min_d \sum_{p \in F} |I(q+p) - I'(q + de_1 + p)|$$

- SSD: Sum of Squared Differences

$$d(q) = \arg\min_d \sum_{p \in F} (I(q+p) - I'(q + de_1 + p))^2$$

- CSSD: Centered Sum of Squared Differences

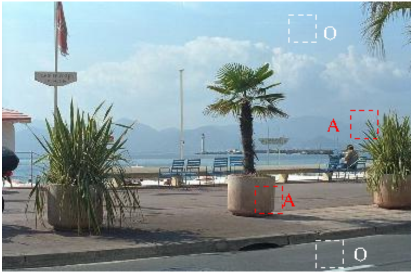
$$d(q) = \arg\min_d \sum_{p \in F} (I(q+p) - \bar{I}_F - I'(q + de_1 + p) + \bar{I}'_F)^2$$

- NCC: Normalized Cross-Correlation

$$d(q) = \arg\max_d \frac{\sum_{p \in F} (I(q+p) - \bar{I}_F)(I'(q + de_1 + p) - \bar{I}'_F)}{\sqrt{\sum (I(q+p) - \bar{I}_F)^2} \sqrt{\sum (I'(q + de_1 + p) - \bar{I}'_F)^2}}$$

Problems of local methods

- Implicit hypothesis: all points of window move with same motion, that is they are in a fronto-parallel plane.
- aperture problem: the context can be too small in certain regions, lack of information.
- adherence problem: intensity discontinuities influence strongly the estimated disparity and if it corresponds with a depth discontinuity, we have a tendency to dilate the front object.



- O: aperture problem
- A: adherence problem

Example: seeds expansion

- We rely on best found distances and we put them in a priority queue (seeds)
- We pop the best seed G from the queue, we compute for neighbors the best disparity between $d(G)-1$, $d(G)$, and $d(G)+1$ and we push them in the queue.

Right image

Example: seeds expansion

- We rely on best found distances and we put them in a priority queue (seeds)
- We pop the best seed G from the queue, we compute for neighbors the best disparity between $d(G) - 1$, $d(G)$, and $d(G) + 1$ and we push them in the queue.

Left image

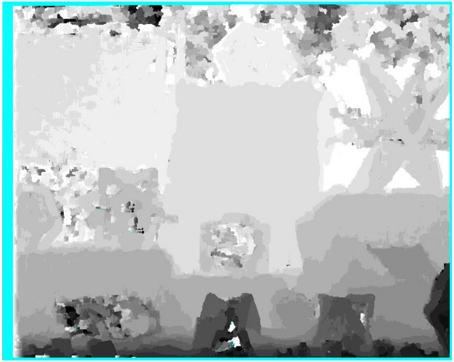
Example: seeds expansion

- We rely on best found distances and we put them in a priority queue (seeds)
- We pop the best seed G from the queue, we compute for neighbors the best disparity between $d(G) - 1$, $d(G)$, and $d(G) + 1$ and we push them in the queue.

Seeds

Example: seeds expansion

- We rely on best found distances and we put them in a priority queue (seeds)
- We pop the best seed G from the queue, we compute for neighbors the best disparity between $d(G) - 1$, $d(G)$, and $d(G) + 1$ and we push them in the queue.



Seeds expansion

Example: seeds expansion

- We rely on best found distances and we put them in a priority queue (seeds)
- We pop the best seed G from the queue, we compute for neighbors the best disparity between $d(G) - 1$, $d(G)$, and $d(G) + 1$ and we push them in the queue.

Left image

Conclusion

- We can get back to the canonical situation by epipolar rectification. Limit: when epipoles are in the image, standard methods are not adapted.
- For disparity map computation, there are many choices:
  1. Size and shape of window?
  2. Which distance?
  3. Filtering of disparity map to reject uncertain disparities?
- You will see next session a *global* method for disparity computation
- Very active domain of research, >100 methods tested at `http://vision.middlebury.edu/stereo/`

Practical session: Disparity map computation by propagation of seeds

Objective: Compute the disparity map associated to a pair of images. We start from high confidence points (seeds), then expand by supposing that the disparity map is regular.

- Get initial program from the website.
- Compute disparity map from image 1 to 2 of all points by highest NCC score.
- Keep only disparity where NCC is sufficiently high (0.9), put them as seeds in a `std::priority_queue`.
- While queue is not empty:
  1. Pop P, the top of the queue.
  2. For each 8-neighbor Q of P having no valid disparity, set d_Q by highest NCC score among $d_P - 1$, d_P, and $d_P + 1$.
  3. Push Q in queue.